1985

Level Curve Concept for Facility Planning Using the Apple II Graphics Tablet

Craig E. Kirkland
University of Central Florida

LEVEL CURVE CONCEPT FOR FACILITY PLANNING
USING THE APPLE II GRAPHICS TABLET

BY

CRAIG EDWARD KIRKLAND
B.S., The Pennsylvania State University, 1979

RESEARCH REPORT

Submitted in partial fulfillment of the requirements
for the degree of Master of Science
in the Graduate Studies Program of the College of Engineering
University of Central Florida
Orlando, Florida

Spring Term
1985

ABSTRACT

The purpose of this effort is to demonstrate the feasibility of using a low cost microcomputer graphics system to assist the practicing engineer in solving facility planning problems. In many cases the lower resolution of the microcomputer system may be adequate and a better choice than a more expensive system.

A demonstration microcomputer graphics system was developed using the Apple II Plus computer and an Apple Graphics Tablet. The facility planning problem chosen was single-facility location using the level curve concept in the rectilinear movement case. Two separate programs were written. The first was used to define the layout of a facility for the computer and the second to draw the level curves.

ACKNOWLEDGEMENTS

I would like to thank all of my family and friends for keeping me calm when the going got tough, especially, Peter Dawson.

I would like to give a special thanks to Joseph Bartlett for his advice and for the use of his computer.

I would also like to thank Leighton Smith for his immense help as my advisor and friend.

Most of all I would like to thank my father, Ron Kirkland, for his support and friendship.

TABLE OF CONTENTS

Appendices

LIST OF TABLES

LIST OF FIGURES

CHAPTER I

INTRODUCTION

With more and more tools becoming available, the practicing engineer must choose carefully what is actually needed to increase productivity. The microcomputer is one of these tools.

It must be kept in mind that the computer is only a tool. Computers are not inherently intelligent and are only as good as their programmers and users. Once a computer has been programmed properly it can perform repetitive calculations and tasks much faster and more reliably than a human. However, if the program is unreliable, the computer is unreliable.

In the area of facility planning a computer can be very useful, especially if it has graphics capability. There are many useful quantitative techniques for facility planning such as linear programming, single and multiple facility location algorithms, and quadratic assignment location algorithms (Francis and White 1974). Many of these are very difficult to execute manually and are, therefore, amenable to solution by computer systems. Some useful layout programs are available on large mainframe computers (Apple 1977). Can any of these algorithms be done on a small system with a lower sophistication of graphics? To answer this question it is necessary to determine what kind of output is needed and how that output will be used (Anderson 1982).

Although quantitative factors are regarded as of primary importance, "qualitative, as well as quantitative, factors should be considered in solving facility layout and location problems" (Francis and White 1974). In many cases an exact answer is not needed and would not be cost effective. The quantitative result is used as a benchmark for further refinements. The microcomputer graphics system can give good results at a much lower price than a more sophisticated system.

An Apple II Plus microcomputer was used with a graphics tablet in order to demonstrate the feasibility of a low cost graphics system. The level curve approach to single facility location was chosen as the quantitative technique to be used. The Apple was chosen because it was available and has very good screen resolution for the price of the system. It also has the support software in BASIC to do the graphics that are needed. The level curve approach was chosen because it can be useful in facility planning but is very tedious to work manually (Moore 1962).

Two programs were written. One was designed to be a master program that defined the layout of the facility under study. The other program uses the information from the master program to draw level curves. In this way another program can be written to execute an algorithm using the information provided by the master program.

CHAPTER II

SINGLE-FACILITY LOCATION: THE LEVEL CURVE CONCEPT

In a single-facility location problem a new facility or machine is to be located with respect to a number of existing facilities or machines. The location should be that which minimizes the total transportation cost of the operation. One useful method is rectilinear-distance location using level curves (Francis and White 1974).

Rectilinear-distance location makes four basic assumptions (Apple 1977): first, it is undesirable to move any existing facilities or machines; second, material flow between two facilities follows a path along two orthogonal axes; third, costs are directly proportional to the distance moved; last, it is sufficient to describe the location of all equipment at a station by defining a point on a layout schematic.

Given these constraints, the cost function of the rectilinear-distance location problem is

$$f(x,y) = \sum_{i=1}^{m} w[i] \; (|x-a[i]| + |y-b[i]|) \qquad (1)$$

Where f(x,y) is the total cost of placing the facility at point (x,y);

m is the total number of existing facilities; (a[i],b[i]) is the (x,y) coordinate of facility i; and w[i] is the cost per unit travel between the new facility and facility i.

If we minimize Equation 1 across all values of x and y, we will obtain the most cost effective location to place the new facility. This location can be a point, a line, or a rectangular area (Apple 1977). It can be shown (Francis and White 1974) that this minimum region will be bounded by an existing x coordinate(s) and an existing y coordinate(s). Also, the optimum coordinates will be at the median location. The median location is that location where no more than one-half the item movement is to the left of (below) the new facility location and no more than one-half the item movement is to the right of (above) the new facility (Francis and White 1974).

For example, assume that a warehouse is to supply the six facilities located at (4,4), (4,10), (6,5), (10,5), (10,9), and (12,3), and that these locations require 4, 4, 2, 3, 5, and 6 trips per day, respectively, the minimum point can be determined by using the data in Tables 1 and 2 as (10,5). The x median location is 10, because it is at this coordinate that the cumulative weight first exceeds half the total weights. The y median location is the interval {5,5} or 5, because at 5 the cumulative weight is equal to exactly one-half the total weights making the median location equal to the interval from 5 to the next coordinate. Since the next coordinate is also 5 the interval becomes just the single coordinate 5.

TABLE 1

MINIMUM X COORDINATE SOLUTION

Existing Facility	X Coordinate Value a[i]	Weight w[i]	Cumulative Weight
A	4	4	4
B	4	4	8
C	6	2	1Ø<12
D	1Ø	3	13>12
E	1Ø	5	18
F	12	6	24

Minimum X Coordinate = 1Ø

TABLE 2

MINIMUM Y COORDINATE SOLUTION

Existing Facility	Y Coordinate Value b[i]	Weight w[i]	Cumulative Weight
F	3	6	6
A	4	4	1Ø
C	5	2	12=12
D	5	3	15>12
E	9	5	2Ø
B	1Ø	4	24

Minimum Y Coordinate = {5,5} or 5

Once this location is determined it is possible that it does not fit the needs of the situation. In this particular example the desired location is already occupied. Or it may be appropriate to compare other possible locations. This requires cost comparison of different locations by using an algorithm such as level curves (also called iso-cost lines or contour lines.)

The following is one method for the manual construction of level curves (Francis and White 1974). In Chapter VI this method will be used in a slightly modified way which is more applicable to computers. Figure 1 should be consulted in stepping through the procedure.

1. Plot the points (a[i],b{i]), . . . ,(a[m],b[m]) of the existing facilities, and draw perpendicular lines (parallel to the x and y axes) through each point.

2. Number the vertical lines 1,2,. . .,p from left to right, and the horizontal lines 1,2,. . .,q from bottom to top.

3. Call the x intercept of the jth vertical line c[j] and the y intercept of the ith horizontal line d[i]. Denote the region delimited by the vertical lines j and j+1 and horizontal lines i and i+1 by [i,j]. In order that all regions are numbered, assume a vertical line Ø to the left of line 1, a vertical line p+1 to the right of line p, a horizontal line Ø below horizontal line 1, and a horizontal line q+1 above horizontal line q.

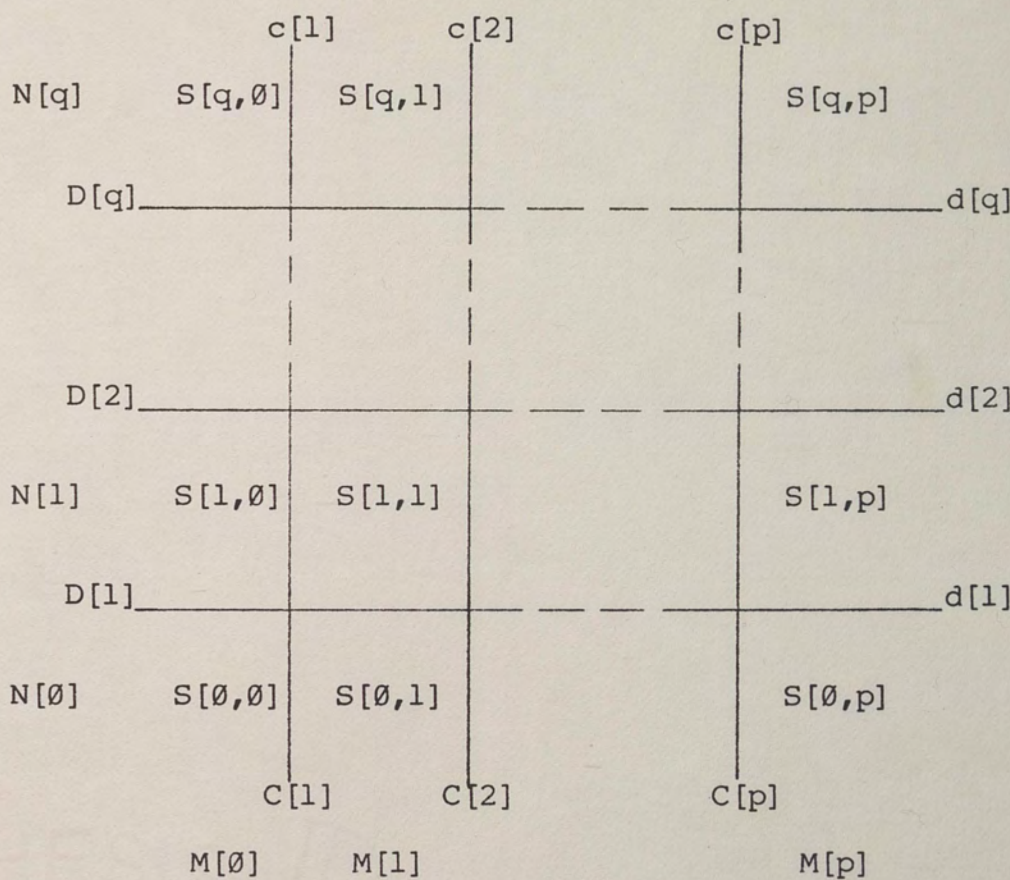

Figure 1. Construction Required to Draw Level Curves

4. Let C[j] and D[j] be the sums of the weights associated with vertical line j and horizontal line i, respectively, and place them as indicated in Figure 1.

5. Compute the numbers

$$M[\emptyset]=-\sum_{j=1}^{p} C[j] \qquad\qquad N[\emptyset]=-\sum_{i=1}^{q} D[i]$$

$$\begin{array}{ll} M[1]=M[\emptyset]+2C[1] & \\ & (2) \\ M[2]=M[1]+2C[2] & \\ \vdots & \\ M[p]=M[p-1]+2C[p]=-M[\emptyset] & \end{array} \qquad \begin{array}{ll} N[1]=N[\emptyset]+2D[1] & \\ & (3) \\ N[2]=N[1]+2D[2] & \\ \vdots & \\ N[q]=N[q-1]+2D[q]=-N[\emptyset] & \end{array}$$

and place them as shown in Figure 1. The equations to find the M values will be referred to as Equation 2, and the equations to find the N values will be referred to as Equation 3.

6. Calculate the slope S[i,j] of any level curve through the region [i,j] as follows :

$$S[i,j]=-M[j]/N[i] \qquad (4)$$

When N[i] is zero, the level curve through the region is vertical.

The calculations to construct level curves have now been completed. Figure 2 illustrates the values obtained for these calculations using the values of the example. Now level curves may be drawn by starting at any point (x,y) as long as (x,y) is not a minimum point. Determination of the minimum point region has been explained using median location. Francis and White (1974) recommend another

		c[1] 4	c[2] 6	c[3] 1Ø	c[4] 12	
N[5] 24	S[5,Ø] 1	S[5,1] 1/3	S[5,2] 1/6	S[5,3] −1/2	S[5,4] −1	
D[5] 4						d[5] 1Ø
N[4] 16	S[4,Ø] 3/2	S[4,1] 1/2	S[4,2] 1/4	S[4,3] −3/4	S[4,4] −3/2	
D[4] 5						d[4] 9
N[3] 6	S[3,Ø] 4	S[3,1] 4/3	S[3,2] 2/3	S[3,3] −2	S[3,4] −4	
D[3] 5						d[3] 5
N[2] −4	S[2,Ø] −6	S[2,1] −2	S[2,2] −1	S[2,3] 3	S[2,4] 6	
D[2] 4						d[2] 4
N[1] −12	S[1,Ø] −2	S[1,1] −2/3	S[1,2] −1/3	S[1,3] 1	S[1,4] 2	
D[1] 6						d[1] 3
N[Ø] −24	S[Ø,Ø] −1	S[Ø,1] −1/3	S[Ø,2] −1/6	S[Ø,3] 1/2	S[Ø,4] 1	
		C[1] 8	C[2] 2	C[3] 8	C[4] 6	
	M[Ø] −24	M[1] −8	M[2] −4	M[3] 12	M[4] 24	

Figure 2. Data Needed for the Construction of Level Curves for Points in Tables 1 and 2.

method using the M and N values calculated in order to draw level curves (see Table 3). Since these values are already calculated and using them to find the minimum point region is relatively simple, this becomes the prefered method for determination by the computer. Both methods are essentially the same and give the same results.

If a level curve were started at point (4,5), the level curve could be drawn using the data in Table 4. These values are very tedious to calculate. Hence, manual calculation of the values can be expected to contain errors. A computer is much better at doing these type of calculations (Fitts 1951). Recall that every point along this level curve or iso-cost line gives the same cost after implementation using the supplied data. This level curve is shown in Figure 3.

TABLE 3

MINIMUM POINT DETERMINATION USING THE M AND N VALUES

M[j-1]	M[j]	N[i-1]	N[i]	X	Y
<Ø	>Ø	<Ø	>Ø	c[j]	d[i]
<Ø	Ø	<Ø	>Ø	{c[j],c[j+1]}*	d[i]
<Ø	>Ø	<Ø	Ø	c[j]	{d[i],d[i+1]}
<Ø	Ø	<Ø	Ø	{c[j],c[j+1]}	{d[i],d[i+1]}

* { } indicate a range which includes the endpoints.

TABLE 4

DATA FOR LEVEL CURVE STARTING AT (4,5)

Starting Point	Region	Slope	Ending Point	Next Region
4,5	3,1	4/3	6,7 2/3	3,2
6,7 2/3	3,2	2/3	8,9	4,2
8,9	4,2	1/4	10,9 1/2	4,3
10,9 1/2	4,3	-3/4	10 2/3,9	3,3
10 2/3,9	3,3	-2	12,6 1/3	3,4
12,6 1/3	3,4	-4	12 1/3,5	2,4
12 1/3,5	2,4	6	12 1/6,4	1,4
12 1/6,4	1,4	2	12,3 2/3	1,3
12,3 2/3	1,3	1	11 1/3,3	0,3
11 1/3,3	0,3	1/2	10,2 1/3	0,2
10,2 1/3	0,2	-1/6	6,3	1,1
6,3	1,1	-2/3	4 1/2,4	2,1
4 1/2,4	2,1	-2	4,5	DONE

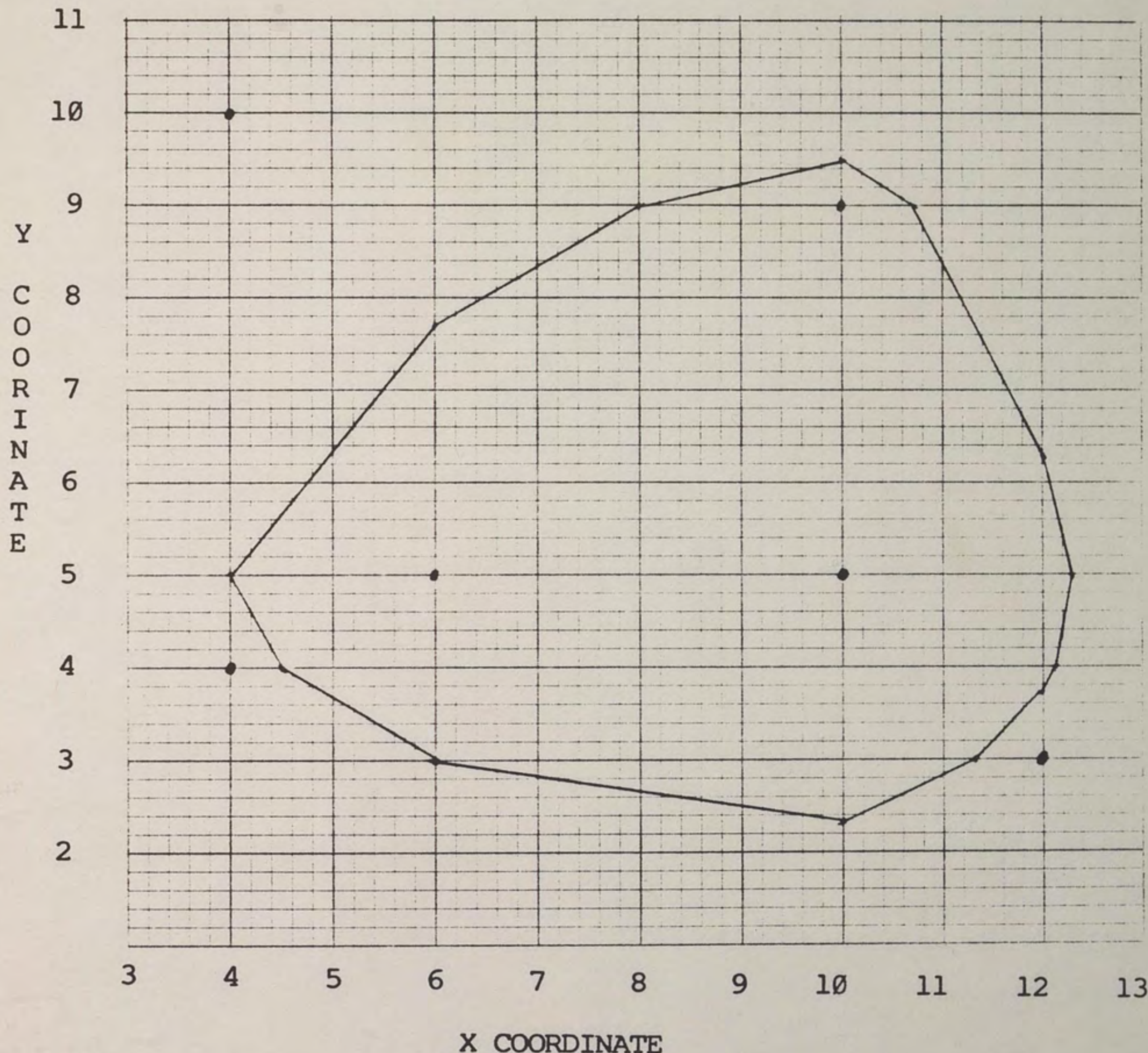

Figure 3. An Example of a Level Curve

CHAPTER III

EQUIPMENT

An Apple II Plus microcomputer with 48K of RAM and two disk drives was used for program development and implementation. One disk drive was found to be sufficient, but a two-drive system would be preferable in order to allow for easier disk backup and the use of other programs requiring the extra storage. The system also included an Amdek Color-I color monitor and the Apple II Graphics Tablet.

In this application the color monitor was not necessary, as no color was used. A monochrome monitor would have been sufficient. However, it is advantageous to have color capability when using any graphics system (Foley and Van Dam 1982). Any monitor used in conjunction with the graphics tablet should be checked for electromagnetic interference (EMI) problems (Apple Computer 1979).

There are five main types of graphics interfaces (Foley and Van Dam 1982). One type is the light pen with which the user directly references points on the display screen by pointing to the screen with a special pen. A second type is the touch screen which the user touches with his finger to indicate a point on the screen. A third is the mouse which the user moves on a flat surface while watching a screen position indicator on the screen. When the screen position indicator is in the proper place the user presses a button. The fourth type is the joystick which works in much the same way as the mouse, but the actuation is slightly different. To move the screen position indicator, the joystick is pushed or pulled in the desired direction. The fifth type, which is what was used in this case, is a graphics tablet.

A graphics tablet functions somewhat like a light pen. It uses a pen which is connected by a wire to the computer. Instead of referencing the display screen, however, it indicates the pen position on the tablet. The tablet is a thin rectangular board, also connected by wires to the computer. The pen has a switch, usually at the tip, which may be pressed to indicate that the current tablet position should be received by the computer.

While the graphics tablet can perform many of the functions of the other devices, it can be less convenient. It is a separate device. The other devices refer directly to the screen coordinates; the tablet coordinates must be converted to screen coordinates by the computer.

The advantage of the tablet is in its separate nature. It lies flat, and is much easier to use for drawing purposes. The tablet also has a much finer resolution than the display screen in most cases. In the case of the Apple, the display screen resolution is 280 X 192 versus a resolution of more than 6000 X 6000 for the Apple II Graphics Tablet.

CHAPTER IV

SOFTWARE DESIGN

All programs are written in Applesoft BASIC and run under DOS 3.3, the Apple disk operating system. The software in this implementation consists of two programs which communicate by means of disk files. The first, BUILD FAC, is loaded to begin. The second, DRAW LEVEL CURVES, is executed by BUILD FAC and returns to BUILD FAC when the user is done.

Figure 4 illustrates the memory map of the Apple II computer (Myers 1982). The Apple II can contain up to 64K of addressable memory. The lower 2K and upper approximately 26.5K of memory are used by the system. (In common computer terminology,1K is defined to be 1024 bytes of memory). This leaves about 35.5K to be used by the program. Referring to the figure, notice that page one of high resolution graphics is located in memory from 8K (8192) to 16K (16384) and page two of high resolution graphics is located in memory from 16K (16384) to 24K (24576). If both pages are used, only 19.5K will be left for the program and data. If a program is large enough, it will expand into graphics page one, making that page unusable. Actually the page may be used for graphics but it could destroy the program. Page two could be used to allow the program to be larger, but this does not allow easy usage of the mixed text graphics mode. In this mixed mode text may be printed on the bottom four lines of the screen, while graphics is on

ADDRESS HEX	ADDRESS DECIMAL	DESCRIPTION
FFFF	65535	
		SYSTEM USAGE
CØØØ	49152	
		DISK OPERATING SYSTEM
96ØØ	384ØØ	
		STRINGS GENERAL USAGE
6ØØØ	24576	
		PAGE 2 OF HI RES GRAPHICS
4ØØØ	16384	
		PAGE 1 OF HI RES GRAPHICS
2ØØØ	8192	
		GENERAL USAGE VARIABLES PROGRAM
Ø8ØØ	2Ø48	
		TEXT PAGE 1
Ø4ØØ	1Ø24	
		SYSTEM USAGE
ØØØØ	Ø	

Figure 4. Apple II Memory Map

the top twenty lines. Due to the advantage of using page one over using page two, the application was split into these two programs in order that it would use less memory.

Another reason for splitting the application was to isolate the program that saves the layouts. This program can then be used with other algorithms in addition the level curve concept. This is in support of the general application concept discussed in Chapter I. The other algorithms could all be separate programs executed like the level curve program.

Chapter V describes the program BUILD FAC, which is used to obtain information about the specific facility. Chapter VI describes the program DRAW LEVEL CURVES, which obtains information from BUILD FAC and then draws level curves.

CHAPTER V

BUILD FAC PROGRAM

This program allows the user to define all relevant locations on a layout and the weights associated with each point. These weights are relative to the point to be entered in the level curve program. Appendix B contains the program listing.

The program is menu driven. "Menu selection requires little or no user training and has the advantage that users may be informed about the range of system features" (Shneiderman 1980). There are nine menu selections, as shown in Figure 5. The purpose and operation of each selection will be described in detail from a system level as opposed to a coding level. Any specific details on code can be found in Appendix B.

Defining a New Facility

The first selection, NEW FACILITY, is chosen by entering an "N" on the keyboard. The user has indicated that a new layout will be defined. A message is immediately displayed to remind the user to attach a schematic of the layout to the graphics tablet. It is important that this is done before proceeding. The layout schematic should be drawn to scale, and the stations defined therein should have a point which is designated as the x,y position of the station.

MENU

N	NEW FACILITY
A	ADD POINTS
D	DELETE POINTS
W	WEIGHTS CHANGE
S	SAVE FACILITY TO DISK
L	LOAD FACILITY FROM DISK
C	CATALOG OF DISK
R	RUN LEVEL CURVE PROGRAM
Q	QUIT

ENTER LETTER OF CHOICE : __

Figure 5. Menu For BUILD FAC Program

The user gives the facility a name, a unit of measure, and the overall size of the facility in the given unit. All facilities are assumed to be of a rectangular nature. An example would be a name of "Warehouse," a unit of "feet," and a size of "1000 feet by 500 feet." For best results, the x direction should be the larger of the two dimensions. In this case that would be 1000 feet.

The screen width (x) and the length (y) of the facility are then calculated using the actual width and length. Note that the maximum screen width (x) is 280 screen units and the maximum screen length (y) is 160 units in mixed text graphics mode. The bottom of the screen is a 40 by 4 character text field. Equations 4 through 7 are used to maximize the screen area used while keeping the proper scaling. If the ratio of the x dimension to the y dimension is less than or equal to one,

$$XL=(159*SC)*XF/YF \qquad (4)$$

$$YL=159 \qquad (5)$$

where XL is the screen width minus one, SC is a scaling factor, XF is the "real world" width, YF is the "real world" length, and YL is the screen length minus one. SC is needed due to the fact that one x unit is not the same length as one y unit on the screen (Myers 1982).

If the ratio is greater than one and the result of Equation 4 is less than 280, then Equations 4 and 5 are used again. If, however, the result of Equation 4 is greater than 280, the following equations are used:

$$XL=279 \quad (6)$$

$$YL=(279*YF/XF)/SC \quad (7)$$

Now that the "real world" coordinates and the screen coordinates of the facility have been determined, the graphics tablet coordinates for the facility must be determined. The computer reminds the user to firmly secure a layout schematic of the facility in question to the tablet by tape or other means. The user is prompted to press the tablet pen down at the upper left corner of the layout schematic. Once this is done the program prompts the user to press the pen down at the lower right corner of the layout schematic. Note that no check is made that the schematic is drawn to scale or that the user has attached the schematic parallel to the tablet axes. It is is left to the user to ensure that these conditions have been met.

Thus the facility has been defined except for the location of the stations relevant to the problem. This can be done by adding the locations via the tablet. The ADD POINTS routine is entered at this stage.

Adding Points

The selection, ADD POINTS, is chosen by entering an "A" from the keyboard. It is also automatically entered after defining a new facility, loading a saved facility, or deleting a point. The purpose of the routine is to define locations in the facility and give these locations weights.

Before beginning, the routine checks to see if a facility is defined. If not, the NEW FACILITY routine is called. If a facility is defined, the computer requests the user to begin adding points. Points are added by pressing the tablet pen down on the graphics tablet within the layout schematic. When all the points are entered, the user signifies this by pressing the pen down outside of the layout schematic.

When the pen is pressed down on the tablet, a letter is drawn on the screen at the corresponding point. An arbitrary limit of eleven points was chosen allowing for points A through K to be entered. The letters are drawn on the screen using the shape table technique (Myers 1982) explained in Appendix D.

If eleven points are already defined upon entering this routine, the computer will return to the main menu. If the user attempts to exit this routine before entering a point when no points have been defined, such is not allowed until at least one point is defined.

Deleting Points

The selection, DELETE POINTS, is chosen by entering a "D" at the keyboard. The user has indicated a desire to remove a previously defined point. If no facility is defined at this stage, the program returns to the main menu.

The current facility is displayed, and the user is requested to enter the letter of the point to remove or "Q" to return to the main menu. If an invalid letter is entered, the program prompts the user to try another letter.

When a valid letter is entered, the user is prompted to verify that this is the point to delete. If anything but a "Y" is entered, the routine goes to its beginning. Otherwise, the point is removed and the other points are consolidated. For example, if points A, B, C, and D are defined and B is deleted, A remains A; C becomes B; D becomes C. Although the former C is now called B, it still has its former weight and location.

After a point is deleted, the routine redraws the facility and automatically enters the ADD POINTS routine. Hence only one point may be deleted at a time. Since the main use of this routine is to remove a point that was inadvertently added in the ADD POINTS routine, returning to ADD POINTS allows a rapid correction of such errors.

Changing Weights

The selection, WEIGHTS CHANGE, is chosen by entering a "W" at the keyboard. The user has indicated a desire to change the weight associated with a point or points. If no facility is defined at this stage, the program returns to the main menu.

The data for the current facility is displayed including the point label, location, and weight. The user is requested to indicate which point by letter will have its weight changed. If the letter is invalid, the routine will return to the main menu. If the letter is valid, the routine asks for the new weight of that point. The weight may be any positive number.

It can be seen that the user may enter this routine merely to review the current data on a facility without the intention of changing a weight. As stated, the user may return to the main menu by entering any invalid letter.

Saving a Facility

The selection, SAVE FACILITY TO DISK, is chosen by entering an "S" at the keyboard. The user has indicated a desire to save the current facility information to a file on the disk. If there is no active facility, the routine returns immediately to the main menu.

This routine deletes any existing file with the name of the current facility and creates a new file with the name of the current facility. The information in the file includes the number of points, the size of the facility in real units, the definition of real units, the tablet coordinates of the upper left and lower right corners of the layout schematic, the screen dimensions, and the data associated with each point.

Once the file is complete, the routine returns to the calling routine. The routines that call this routine are the main menu and RUN LEVEL CURVE PROGRAM.

Loading a Saved Facility

The selection, LOAD FACILITY FROM DISK, is chosen by entering an "L" at the keyboard. The user has indicated that a file containing information about a facility will be loaded from the disk and will define the active facility. If there is currently an active facility, the user is asked if he wants to continue as it will be destroyed.

The computer prompts the user for a facility name. If the user gives an invalid file name or a nonexistant file name, the program will terminate. Nothing is lost as a result of this, and the program may be restarted.

Assuming that a valid name has been entered, the file is loaded. Then, the data must be aligned with respect to the graphics tablet. This is done by executing the part of the NEW FACILITY routine which aligns the layout schematic on the tablet. Given the alignment information, the different points are realigned in the facility as follows:

$$PO(\emptyset,I)=PO(\emptyset,I)-(XQ-X1) \quad (8)$$

$$PO(1,I)=PO(1,I)-(YQ-Y1) \quad (9)$$

where PO(Ø,I) is the x coordinate of point I, PO(1,I) is the y coordinate of point I, XQ is the previous graphics tablet x coordinate of the upper left corner of the layout schematic, X1 is the new x coordinate, YQ is the previous graphics tablet y coordinate of the upper left corner of the layout schematic, and Y1 is the new y coordinate. Next the facility is drawn on the graphics screen and the ADD POINTS routine is entered.

Obtaining a List of Disk Files

The selection, CATALOG OF DISK, is chosen by entering a "C" at the keyboard. A list of the files on the disk will be displayed until any key is pressed. When a key is pressed, the program will return to the main menu.

Executing the Level Curve Program

The selection, RUN LEVEL CURVE PROGRAM, is chosen by entering an "R" at the keyboard. The user has indicated that a facility has been defined and that it will be the facility used by the level curve program. If there is no active facility, the routine returns to the main menu.

This routine calls the routine, SAVE FACILITY TO DISK, to save the data for the current facility. Then, a file called TEMP is created on the disk. This file contains the name of the current facility in order that the level curve program will know which disk file to access. Then the level curve program is executed.

Exiting From the Program

The selection, QUIT, is chosen by entering a "Q" at the keyboard. The user has indicated that he is finished and he will be placed in the command mode of Applesoft BASIC.

CHAPTER VI

DRAW LEVEL CURVES PROGRAM

This program is run by the BUILD FAC program in order to draw level curves. It uses data supplied by that program in order to do this. When the user is done drawing level curves with the current data set, this program is exited and the BUILD FAC program is executed. This program will be described by referring to the flow chart in Figure 6. Level curves will be constructed as in Chapter II unless otherwise noted. Appendix C contains a listing of this program.

Initialization

Before beginning, data areas must be designated. The variables are assigned an area beginning at memory location 24576. This is done to prevent interference with the graphics pages (see Figure 5). Normally, variables are stored in memory immediately after the program. The shape table containing the alphanumeric images is loaded at location 33ØØØ. BASIC initializes all variables to zero before running the program.

Data Input

The program now needs data supplied by BUILD FAC. BUILD FAC has saved a file called TEMP, which contains the name of the file with the necessary data. This file is read and the points are converted from tablet coordinates to facility coordinates. As each point is read and

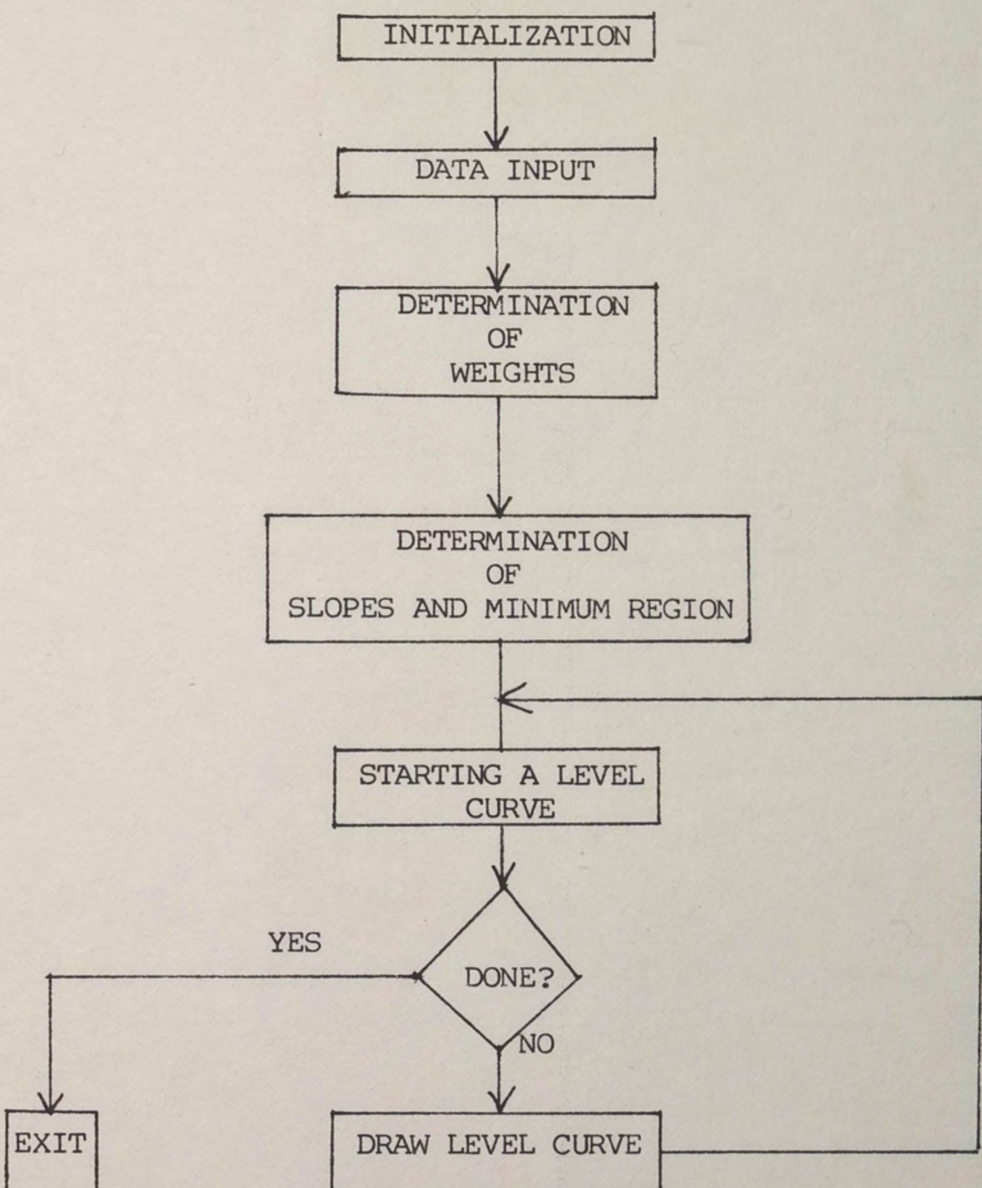

Figure 6. Flowchart For DRAW LEVEL CURVES

converted, it is drawn as a letter on the screen. The conversion is the same as that used in BUILD FAC. After conversion all calculations are done using facility coordinates. All coordinates are converted to screen coordinates before drawing to the screen.

Determination of Weights

After the data for the facility has been read from the disk, weights must be determined. The weights of all the points are summed. Then the minimum and maximum x and y coordinates are determined. Next, the weight of each point is normalized by dividing it by the sum of the weights. Now all weights should add up to one.

The weights and coordinate values of each horizontal and vertical line are then determined. This can either be done by sorting of points or by comparing every point with all possible coordinates from minimum possible to maximum possible point. Sorting of points is faster but requires more memory and coding. Since this determination is done only once each time the program is executed, speed is sacrificed in favor of memory space conservation. Therefore, the comparison of points method is used. The process is the same for both the x and y axes, and therefore, an explanation will be provided for the x coordinates only.

A loop is set up to step integrally from the minimum x coordinate to the maximum x coordinate. At each coordinate, all points have their x coordinate compared to see if it is the same as the current coordinate. If at least one matches then this is the coordinate of one of the vertical lines c[j] described in Chapter II. The weights of that matching point plus any other matching points are added to get the weight C[j] for the vertical line c[j]. A count is kept of all vertical lines plus the information associated with each line. The lines are numbered from one to p where p is the number of lines. A line zero and a line p+1 are added with coordinates -1000 and 1000, respectively. This is done to ensure that all regions are bordered. These numbers are deliberately large so as to represent negative and positive infinity. The determination of horizontal lines using y coordinates is done in the same way. These horizontal and vertical lines border regions which will be identified as in Figure 1.

Determination of Slopes and the Minimum Region

To determine the slopes of the different regions, the M and N values of Figure 1 must be calculated as in the group of Equations 2 and 3. The only difference is that N[0] and M[0] are defined to be negative one because all the weights have been normalized.

As the M and N values are calculated, the value of the sign is calculated. Negative is negative one, zero is zero, and positive is positive one. The number of sign changes determines the minimum point region (see Table 3). The minimum point region information is stored for later use and displayed on the screen.

Now that the M and N values are calculated, the slope S for each region can be calculated using Equation 4. If N is equal to zero, the slope is set to zero and a flag is set to negative one for that slope. All other slopes have a flag of zero.

Starting a Level Curve

The user is asked to select a point to begin a level curve. If the pen is pressed down on the tablet within the facility boundaries, a level curve is drawn starting at that point. If the pen is outside the boundaries, the user is given a choice of exiting to the BUILD FAC program or selecting another point to begin a level curve.

Once the point is entered and converted to screen coordinates, it is checked to make sure that it is not within the minimum region. If it is, the user must choose another point. If not, the computer determines the starting region [i,j] of the point.

Note that once the calculations have been made, drawing level curves is easily done by a human. Although such a task can be time consuming, a person can determine what region a point is in and where to draw the next line merely by inspection. Computers, on the other hand, must keep track of where they have been and where they currently are to determine where next they must go. Once a computer has been programmed properly, however, it is much faster and more reliable than a person.

To determine what region it is starting in, the computer must address four cases, as indicated in Figure 7. The program starts at c[Ø] and checks the starting x coordinate to find when c[k] is greater than the x coordinate. In this case, that will occur at c[j+1]. Doing the same for the y coordinate will yield d[i+1] as the first greater value. By this, the first determination is that the starting region will be [i,j]. This determination will be correct for point A. This choice is also acceptable for points B and C because a starting region is being selected and both possible regions will contain part of the level curve. Therefore, the choice is arbitrary. However, point D is at a vertex and [i,j] is only one of four possible regions and may not be the proper choice.

The region [i,j] is always a valid starting region for point D if the slope S[i,j] is positive. If S[i,j-1] is not positive and S[i,j] is negative, then [i,j-1] is a valid starting region. If, however, S[i,j] and S[i,j-1] are equal to zero and S[i-1,j] is negative, the region chosen is [i,j-1]. If S[i-1,j] and S[i,j] are zero and S[i,j-1] is negative, the region chosen is [i-1,j]. Once the starting region is determined, it is saved because the first region is also the last region. A level curve always begins and ends in the same region and at the same point, but the computer always draws a complete line in the starting region. When the computer draws the last segment of the level curve, it has returned to the starting region. Therefore, only the information for the starting region is saved or needed.

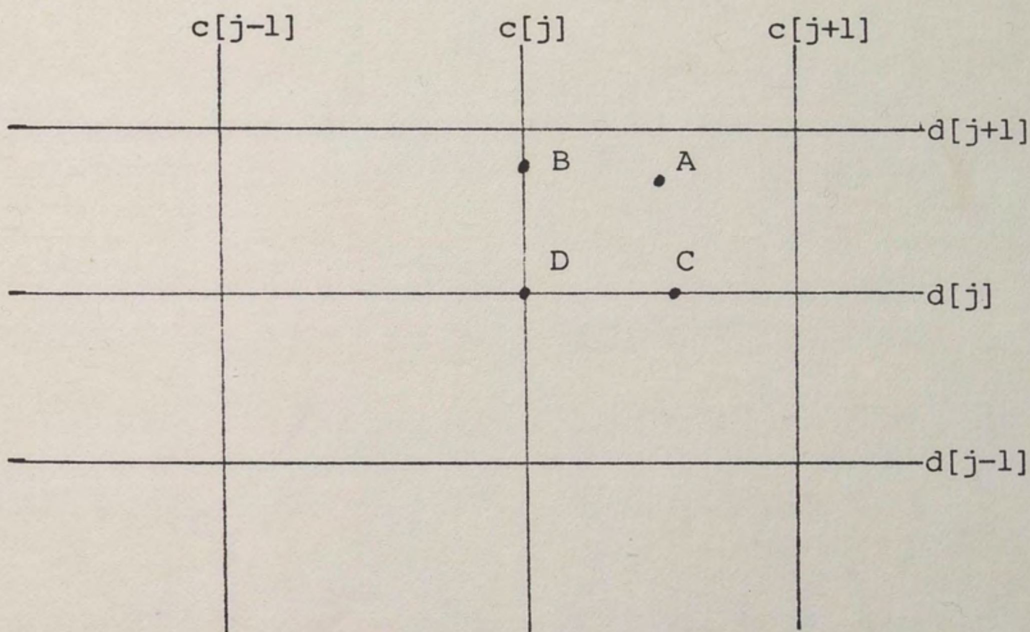

Figure 7. Possible Classes of Starting Points

Drawing the Level Curve

Given the starting point, starting region, and the point-slope equation

$$Y2-Y1=m(X2-X1) \qquad (10)$$

the endpoints of the line through the region can be determined and drawn. Both of the endpoints will have at least the x or y coordinate as an intercept of a vertical or horizontal line. The second endpoint determined becomes the new starting point and the new region must be determined. Lines that do not have both endpoints on the screen are not drawn.

Since at least the x or y coordinate of the starting point is an intercept, only the B, C, and D cases of Figure 7 must be considered in determining the new region. In the case of B or C, there are only two possible regions. These regions can be determined, and the one chosen is the one that was not the previous region. The previous region is the region where part of the curve has just been drawn.

Case D is again more complex. There are four possible vertex points in the previous region as shown in Figure 8. Each vertex point has three possible new regions, since the previous region is not eligible. This discussion assumes that the vertex point is the lower left one, but the analysis is easily converted to any of the other vertex points. The new region will be region 3 unless the sign of region 3 is opposite to that of region 1, the previous region. Then the new region will be

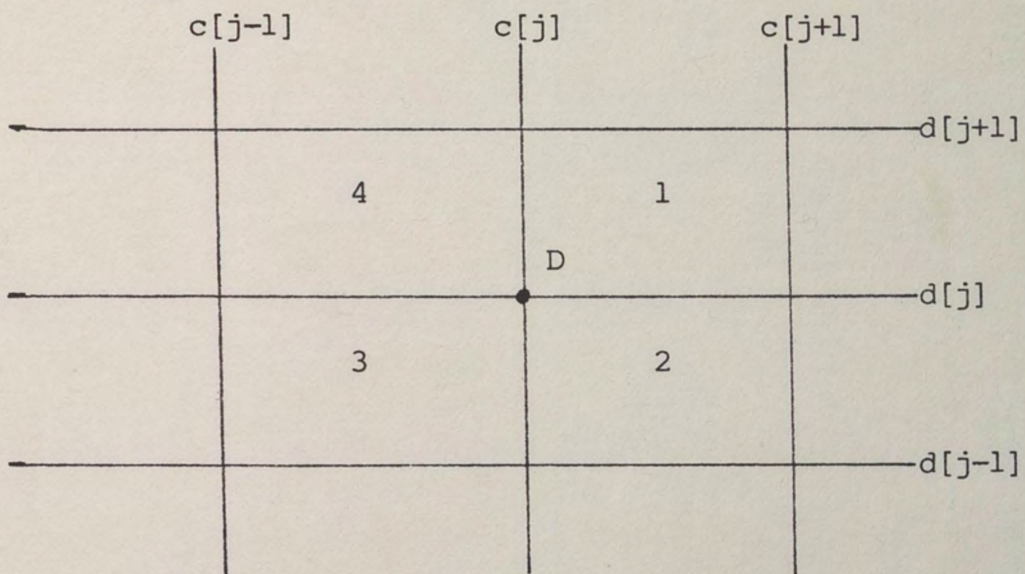

Figure 8. Possible New Regions for Vertex Point D

region 2 if its sign is the same as that of region 3. Otherwise, the new region is region 4. Opposite means positive as opposed to negative in this case. Zero is not an opposite.

The new region is compared to the original starting point region. If they are the same, the level curve is complete and the program asks for another starting point. If not, then this routine continues with the next iteration.

An Example

To test the program the example from Chapter II was done by the computer. Once the data for the facility was entered by the BUILD FAC program, this program was run to draw the level curve.

The data for the level curve is shown in Table 5. Notice that the starting point is at (3.986, 5) and not at (4, 5). This is due to the fact that a person entering a point by the tablet may be slightly off, and the translation from tablet to real coordinates for a starting point does not attempt to resolve this at this time. A comparison of the data of Table 5 to that of Table 4 shows a close correspondence and only a few extra line segments for the computer. These extra segments occur at places where the level curve in the manual version goes through a vertex point such as (6, 3). The computer version did not in this case draw through any vertex points.

Figure 9 is a printer output of the screen contents after this level curve was drawn. This program does not output graphics information to a printer or plotter. The reason for this is that the

TABLE 5

COMPUTER DATA FOR LEVEL CURVE STARTING AT (4,5)

Starting Point	Region	Slope	Ending Point	Next Region
3.986,5	3,Ø	4	4,5.Ø57	3,1
4,5.Ø57	3,1	1.333	6,7.724	3,2
6,7.724	3,2	Ø.667	7.915,1Ø	4,2
7.915,1Ø	4,2	Ø.25	1Ø,9.521	4,3
1Ø,9.521	4,3	-Ø.75	1Ø.695,9	3,3
1Ø.695,9	3,3	-2	12,6.39Ø	3,4
12,6.39Ø	3,4	-4	12.347,5	2,4
12.347,5	2,4	6	12.181,4	1,4
12.181,4	1,4	2	12,3.638	1,3
12,3.638	1,3	1	11.362,3	Ø,3
11.362,3	Ø,3	Ø.5	1Ø,2.319	Ø,2
1Ø,2.319	Ø,2	-Ø.167	6,2.985	Ø,1
6,2.985	Ø,1	-Ø.333	5.957,3	1,1
5.957,3	1,1	-Ø.667	4.457,4	2,1
4.457,4	2,1	-2	4,4.915	2,Ø
4,4.915	2,Ø	-6	3.986,5	DONE

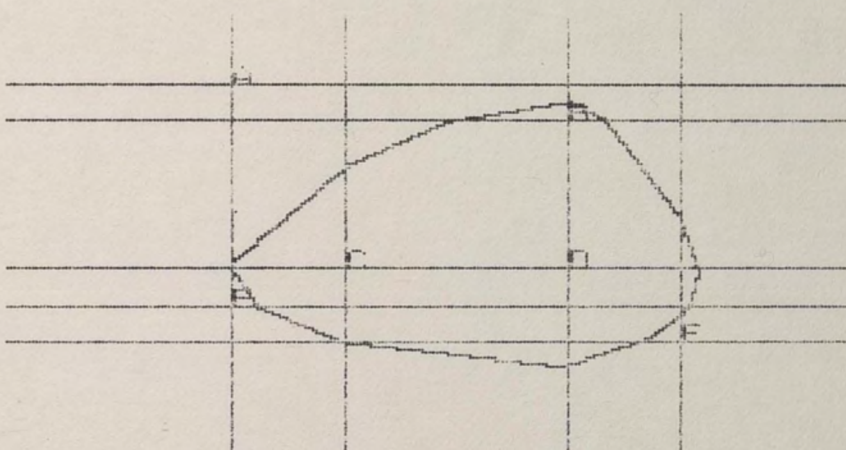

Figure 9. Printer Output of Screen Display.

programming tends to be very hardware dependent. It would have been necessary to write a special software graphics driver to use a hardcopy device and, therefore, was beyond the scope of this project. Appendix E describes the creation of Figure 9.

CHAPTER VII

CONCLUSIONS AND RECOMMENDATIONS

A program has been developed which can be very useful for facility planning and where the cost of the system is very low compared with many of the graphics systems available today. It should be seen that one must determine what is really needed in a graphics system before spending more money than is necessary.

As a tool, the microcomputer graphics system has proven itself to be very effective in many engineering applications. As in using any tool one must keep in mind that a tool is only as good as the person using it. In the level curve approach, certain assumptions have been made. First, it is undesirable to move any existing facilities or machines; second, material flow between two facilities follows a path along two orthogonal axes; third, costs are directly proportional to the distance moved; fourth, it is sufficient to describe the location of all equipment at a station by defining a point on a layout schematic. The engineer must keep in mind whether his model actually fits these assumptions before using this approach.

The system in its present state can be used to compare the cost of different locations for the placement of a new station in a facility. Some of the recommendations for additions to the system are the addition of a hardcopy graphics device, the provision for the output of more cost information, the inclusion of sensitivity analysis, and the addition of more layout programs such as the level curve concept for the euclidean case to use the data provided by the master program.

Hardcopy output is often necessary in a graphics system, but the type of output needed varies with the application and cost of the system. The different devices need different hardware and software interfaces which are beyond the scope of this project. Appendix E illustrates one such interface.

Since this level curve concept is based on cost, more cost information could be provided. Items such as the cost in the minimum region and along the level curve being drawn could be provided. A level curve could be chosen by a percentage over minimum cost as opposed to a starting point.

Sensitivity analysis would be a useful addition to the level curve program. The ability to determine effects of individual stations on the displayed level curve by changing weights and locations would be helpful to an analyst.

Finally, more programs can be written implementing layout algorithms such as the level curve concept for the euclidean case. These programs could use the master program to obtain the data for analysis.

APPENDICES

APPENDIX A

USING THE APPLE II GRAPHICS TABLET

A BASIC program can communicate with the graphics tablet in the same way it communicates with other peripherals using the PRINT and INPUT statements. To decide which devices to print to or input from, the program must execute a PR#S and IN#S statement, respectively. Where S is the actual slot number of the device interface card.

After commands are sent with the print command to configure the tablet, information is ready to be received from the tablet. Input from the tablet consists of three numbers: the x position, the y position, and the tablet status. The status indicates whether the pen is down and has been down, the pen has just been lifted, or the pen was just pressed down. The programs in this study get the position as the pen is pressed down and wait for the pen to be lifted before getting another point.

APPENDIX B

LISTING OF BUILD FAC PROGRAM

```
2  DIM PO(2,15)
5  HGR
6  HOME
7 CH = 11
8 NP = 0
9 SC = 1.16
15  PRINT  CHR$ (4);"PR#3": PRINT
     "D": PRINT  CHR$ (4);"PR#0"
17  TEXT
18 I = 30000
19  PRINT  CHR$ (4);"BLOAD SHAPE
     ALPHANUM,A";I
20  POKE 232, PEEK (43634): POKE
     233, PEEK (43635)
22  TEXT : GOSUB 2500
25  IF A$ = "N" GOTO 490
26  IF A$ = "A" GOTO 600
27  IF A$ = "Q" THEN  TEXT : END

28  IF A$ = "S" THEN  GOSUB 4000
29  IF A$ = "L" GOTO 4200
30  IF A$ = "C" GOTO 4400
31  IF A$ = "R" GOTO 4500
32  IF A$ = "D" GOTO 5000
33  IF A$ = "W" GOTO 5300
35  GOTO 22
490  IF  LEN (F$) = 0 GOTO 500
492  HOME : VTAB 10: PRINT "DO YO
     U WANT TO LOSE THE FACILITY
     GEORGE (Y OR N)";: INPUT "";
     B$
494  IF B$ <  > "Y" GOTO 22
496 F$ = ""
500  REM  NEW FAC
501  HOME : VTAB 4: PRINT "PLEASE
      ATTACH A LAYOUT SCHEMATIC O
     F"
502  PRINT "THE FACILITY IN THE C
     ENTER OF THE TABLET"
503  PRINT : PRINT "BETTER RESULT
     S WILL BE OBTAINED": PRINT "
     IF THE WIDEST PART IS IN THE
      X DIRECTION"
```

```
504  PRINT : PRINT : PRINT : PRINT
     "PRESS ANY KEY WHEN DONE";: GET
     B$
505  IF  LEN (F$) > 0 GOTO 516
506  HOME : VTAB 7: INPUT "FACILI
     TY NAME";F$
507  IF  LEN (F$) = 0 GOTO 506
510  INPUT "FACILITY SIZE UNIT:";
     U$
515  PRINT "FACILITY SIZE IN ";U$
     ;" (X,Y):";: INPUT "";XF,YF
516  IF (XF / YF) > 1 GOTO 518
517 XL = (159 * SC) * XF / YF:YL =
     159: GOTO 520
518 XL =  INT (XF / YF * 159 * SC
     ): IF XL < 280 THEN YL = 159
     : GOTO 520
519 YL =  INT ((YF / XF * 279) /
     SC):XL = 279
520  VTAB 22: HTAB 5: PRINT "PRES
     S PEN DOWN AT UPPER"
522  HTAB 5: PRINT "LEFT CORNER O
     F FACILITY"
523  HGR : HPLOT XL,0 TO XL,YL TO
     0,YL TO 0,0 TO XL,0
525  VTAB 1: GOSUB 2000
527  PRINT  CHR$ (7);
530 X1 = XS:Y1 = YS
535  VTAB 22: HTAB 5: PRINT "PRES
     S PEN DOWN AT LOWER": HTAB 5
     : PRINT "RIGHT CORNER OF FAC
     ILITY"
540  VTAB 1: GOSUB 2000
542  PRINT  CHR$ (7);
545 X2 = XS:Y2 = YS
570  IF A$ = "X" THEN  RETURN
600  REM  ADD POINTS
601  IF  LEN (F$) = 0 GOTO 505
602  FOR I = NP TO 15
604  HOME
605  VTAB 21: HTAB 10: PRINT "ADD
      POINTS"
```

```
610  PRINT "PRESS PEN DOWN OUTSID
     E FACILITY TO EXIT"
620  VTAB 1: GOSUB 2000
622  IF XS < X1 OR YS < Y1 OR XS >
     X2 OR YS > Y2 THEN I = 31: GOTO
     635
625 X = XS:Y = YS:CH = I + 33: GOSUB
     3000
627  HOME
630 PO(0,I) = XS:PO(1,I) = YS
631  VTAB 23: PRINT "WEIGHT AT PO
     INT "; CHR$ (I + 65);" ? ";:
      INPUT " ";PO(2,I)
632 NP = I + 1
635  NEXT I
640  GOTO 22
2000  REM  GENERATE PEN COORDINAT
     ES
2002 C = 0
2005  PRINT  CHR$ (4);"PR#3"
2010  PRINT "M1,P,C"
2015  PRINT  CHR$ (4);"IN#3"
2020  INPUT X,Y,Z
2022  VTAB 1
2025 Z =  ABS (Z): IF Z >  = 10 THEN
     Z = Z - 10
2030  IF Z = 0 OR Z = 2 GOTO 2037

2035  IF C = 0 GOTO 2020
2036  GOTO 2040
2037  IF C = 1 GOTO 2020
2038 XS = X:YS = Y:C = 1: GOTO 20
     20
2040  PRINT  CHR$ (4);"PR#0"
2045  PRINT  CHR$ (4);"IN#0"
2050  RETURN
2500  REM  MENU
2505  HOME : VTAB 2: HTAB 15: PRINT
     "MENU"
2510  HTAB 15: PRINT "----"
2512  PRINT : HTAB 10: PRINT "N
     NEW FACILITY"
```

```
2515  PRINT : HTAB 10: PRINT "A
     ADD POINTS"
2520  PRINT : HTAB 10: PRINT "D
     DELETE POINTS"
2522  PRINT : HTAB 10: PRINT "W
     WEIGHTS CHANGE"
2525  PRINT : HTAB 10: PRINT "S
     SAVE FACILITY TO DISK"
2530  PRINT : HTAB 10: PRINT "L
     LOAD FACILITY FROM DISK"
2535  PRINT : HTAB 10: PRINT "C
     CATALOG OF DISK"
2540  PRINT : HTAB 10: PRINT "R
     RUN ISO-COST LINE PROGRAM"
2550  PRINT : HTAB 10: PRINT "Q
     QUIT"
2555  PRINT : PRINT : HTAB 5
2560  PRINT "ENTER LETTER OF CHOI
     CE:";: GET A$
2562  PRINT
2575  RETURN
3000  REM  DRAW CHAR
3002  REM  TAB COORDS IN X,Y
3004  REM  CHAR IN CH
3006  GOSUB 3010
3007  XDRAW CH AT XC,YC: RETURN
3010 XC = ((X - X1) / (X2 - X1)) *
     XL
3015 YC = ((Y - Y1) / (Y2 - Y1)) *
     YL
3017 XC =  INT (XC + .500):YC =  INT
     (YC + .500)
3030  RETURN
4000  REM  SAVE FILE
4005  IF  LEN (F$) = 0 THEN  RETURN

4010  PRINT  CHR$ (4);"OPEN ";F$
4020  PRINT  CHR$ (4);"DELETE ";F
     $
4030  PRINT  CHR$ (4);"OPEN ";F$
4040  PRINT  CHR$ (4);"WRITE ";F$
```

```
4050  PRINT NP: PRINT XF: PRINT Y
     F: PRINT U$
4052  PRINT X1: PRINT X2: PRINT Y
     1: PRINT Y2: PRINT XL: PRINT
     YL
4060  FOR I = 0 TO (NP - 1)
4070  PRINT PO(0,I): PRINT PO(1,I
     )
4075  PRINT PO(2,I)
4080  NEXT I
4085  PRINT  CHR$ (4);"CLOSE"
4090  RETURN
4200  REM  READ FILE
4205  HOME : VTAB 7: INPUT "FACIL
     ITY TO LOAD ? ";F$
4220  PRINT  CHR$ (4);"OPEN ";F$
4230  PRINT  CHR$ (4);"READ ";F$
4235  INPUT NP,XF,YF,U$,X1,X2,Y1,
     Y2,XL,YL
4240  FOR I = 0 TO (NP - 1)
4250  INPUT PO(0,I),PO(1,I)
4255  INPUT PO(2,I)
4260  NEXT I
4262 XQ = X1:YQ = Y1
4270  PRINT  CHR$ (4);"CLOSE"
4271 A$ = "X": GOSUB 500
4272  FOR I = 0 TO (NP - 1)
4274 PO(0,I) = PO(0,I) - (XQ - X1
     )
4275 PO(1,I) = PO(1,I) - (YQ - Y1
     )
4276  NEXT I
4279  POKE  - 16297,0: POKE  - 16
     304,0: POKE  - 16300,0: POKE
      - 16301,0
4280  FOR I = 0 TO (NP - 1)
4285 X = PO(0,I):Y = PO(1,I):CH =
     I + 33: GOSUB 3000
4287  NEXT I
4290  GOTO 600
4400  REM  CATALOG
4410  PRINT  CHR$ (4);"CATALOG"
```

```
4415  PRINT "     PRESS ANY KEY TO
      CONTINUE....": GET A$
4420  GOTO 22
4500  REM  RUN FG PROG
4501  IF  LEN (F$) = 0 GOTO 22
4502  GOSUB 4000
4505  PRINT  CHR$ (4);"OPEN TEMP"

4510  PRINT  CHR$ (4);"DELETE TEM
     P"
4515  PRINT  CHR$ (4);"OPEN TEMP"

4520  PRINT  CHR$ (4);"WRITE TEMP
     "
4525  PRINT F$
4530  PRINT  CHR$ (4);"CLOSE"
4535  PRINT  CHR$ (4);"RUN DRAW L
     EVEL CURVES"
5000  REM  DELETE POINTS BY LETTE
     R
5005  IF  LEN (F$) = 0 GOTO 5100
5010  PRINT  CHR$ (4);"PR#3"
5015  PRINT "M1,P,C"
5020  PRINT  CHR$ (4);"PR#0"
5022  HOME : VTAB 22: PRINT "ENTE
     R LETTER OF POINT TO DELETE
     "
5025  INPUT "OR Q TO QUIT : ";A$
5030  IF ( ASC (A$) - 65) >  = 0 AND
     ( ASC (A$) - 65) < NP GOTO 5
     035
5032  IF A$ = "Q" GOTO 5100
5034  GOTO 5022
5035  HOME : VTAB 22: PRINT "ARE
     YOU SURE YOU WANT TO;
5036  PRINT "DELETE LOCATION ";A$
     ;: INPUT " (Y OR N)?? ";B$
5040  IF B$ <  > "Y" GOTO 5025
5045  FOR I = ( ASC (A$) - 65) TO
     (NP - 1)
5048  FOR J = 0 TO 2
5050 PO(J,I) = PO(J,I + 1)
5055  NEXT J: NEXT I
```

```
5057 NP = NP - 1
5060  GOTO 4279
5100  GOTO 22
5300  REM  CHANGING WEIGHTS
5305  IF  LEN (F$) = 0 GOTO 22
5310  HOME : PRINT "POINT     LOCA
     TION(X,Y)      WEIGHT"
5315  FOR I = 0 TO NP - 1
5320 B$ =  CHR$ (I + 65): PRINT  SPC(
     2);B$;
5322 X = PO(0,I):Y = PO(1,I): GOSUB
     3010
5323 XC =  INT (XC * XF / XL + .5
     ):YC =  INT (YC * YF / YL +
     .5)
5324 YC = YF - YC
5325 A =  LEN ( STR$ (XC)):A = 5 -
     A: IF A < 0 THEN A = 0:
5330  PRINT  SPC( 8); SPC( A);XC;
     ",";
5335 A =  LEN ( STR$ (YC)):A = 5 -
     A: IF A < 0 THEN A = 0
5340  PRINT  SPC( A);YC; SPC( 5);
     PO(2,I)
5345  NEXT I
5350  VTAB 22: PRINT "ENTER LETTE
     R OF POINT TO BE CHANGED "
5353  INPUT "(ENTER INVALID LETTE
     R TO EXIT";B$
5355 I =  ASC (B$) - 65: IF I < 0
      OR I > (NP - 1) GOTO 22
5360  HOME : VTAB 10: PRINT "OLD
     WEIGHT=";PO(2,I);" AT ";B$;"
      NEW VALUE=";: INPUT "";A
5365  IF A < 0 GOTO 5360
5370 PO(2,I) = A: GOTO 5310
5375  GOTO 22
```

APPENDIX C

LISTING OF DRAW LEVEL CURVES PROGRAM

```
3  LOMEM: 24576
4  DIM XA(1),YA(1)
5  DIM S(1Ø,1Ø,5),PO(1Ø,2),C(1Ø,1
     ),D(1Ø,1)
6  DIM XG(1),YG(1)
7  DIM M(1Ø),N(1Ø)
8  DIM MP(1,1)
9  HCOLOR= 3
1Ø  DIM X9(25),Y9(25)
45 I = 33ØØØ
5Ø  PRINT  CHR$ (4);"BLOAD SHAPE
      ALPHANUM,A";I
55  POKE 232, PEEK (43634): POKE
      233, PEEK (43635)
6Ø  HGR
9Ø MX = 3ØØ:MY = 2ØØ:LX = Ø:LY =
      Ø:MØ = Ø
91 XC = Ø:YC = Ø
95 C(Ø,Ø) =  - 1ØØØ:D(Ø,Ø) =  - 1
      ØØØ
97  GOSUB 65ØØ: REM  INPUT
98  HOME : VTAB 23: HTAB 7: PRINT
      "CALCULATING..";
1ØØ  FOR I = 1 TO 1Ø
12Ø MØ = MØ + PO(I,2)
125  IF MX > PO(I,Ø) THEN MX = PO
      (I,Ø)
126  IF MY > PO(I,1) THEN MY = PO
      (I,1)
13Ø  IF LX < PO(I,Ø) THEN LX = PO
      (I,Ø)
131  IF LY < PO(I,1) THEN LY = PO
      (I,1)
135  NEXT I
137  PRINT ".";
14Ø  FOR I = 1 TO NP:PO(I,2) = PO
      (I,2) / MØ: NEXT I
19Ø J = 1
2ØØ  FOR I = MX TO LX
2Ø5 C(J,Ø) = I:C(J,1) = Ø
21Ø  FOR IX = 1 TO NP
215  IF PO(IX,Ø) = I THEN C(J,1) =
      C(J,1) + PO(IX,2)
```

```
220  NEXT IX
230  IF C(J,1) > 0 THEN J = J + 1

240  NEXT I
241  PRINT ".";
242 C(J,0) = 1000
245 J = J - 1
250 K = 1
255  FOR I = MY TO LY
260 D(K,0) = I:D(K,1) = 0
265  FOR IX = 1 TO NP
270  IF PO(IX,1) = I THEN D(K,1) =
     D(K,1) + PO(IX,2)
275  NEXT IX
280  IF D(K,1) > 0 THEN K = K + 1

285  NEXT I
286  PRINT ".";
287 D(K,0) = 1000
290 K = K - 1
295 M(0) =  - 1
300  FOR I = 1 TO J
305 M(I) = M(I - 1) + 2 * C(I,1)
310  IF  ABS (M(I)) < .01 THEN M(
     I) = 0
311  IF  SGN (M(I - 1)) <  >  SGN
     (M(I)) THEN MP(XC,0) = C(I,0
     ):XC = XC + 1
312  NEXT I
315 N(0) =  - 1
320  FOR I = 1 TO K
325 N(I) = N(I - 1) + 2 * D(I,1)
330  IF  ABS (N(I)) < .01 THEN N(
     I) = 0
331  IF  SGN (N(I - 1)) <  >  SGN
     (N(I)) THEN MP(YC,1) = D(I,0
     ):YC = YC + 1
332  NEXT I
335  GOSUB 2000: REM  MIN PT
340  FOR I = 0 TO J
350  FOR IX = 0 TO K
375  IF N(IX) <  > 0 GOTO 385
```

```
380 S(IX,I,0) = 0:S(IX,I,1) =  -
     1
382  GOTO 390
385 S(IX,I,0) =  - M(I) / N(IX):S
     (IX,I,1) = 0
390  NEXT IX: NEXT I
395  HOME
400  VTAB 21: HTAB 5: PRINT "PRES
     S PEN DOWN AT THE"
402  VTAB 22: HTAB 5: PRINT "CONT
     OUR LINE STARTING LOCATION"
403  VTAB 23: HTAB 7: PRINT "(OUT
     SIDE FACILITY TO EXIT)"
404  GOSUB 6000
405  VTAB 1: GOSUB 7500
406  IF XS < XG(0) OR XS > XG(1) OR
     YS < YG(0) OR YS > YG(1) GOTO
     8000
410  IF XC > 1 GOTO 420
415  IF X <  > MP(0,0) GOTO 440
416  GOTO 425
420  IF X < MP(0,0) OR X > MP(1,0
     ) GOTO 440
425  IF YC > 1 GOTO 435
430  IF Y <  > MP(0,1) GOTO 440
431  GOTO 400
435  IF Y >  = MP(0,1) AND Y <  =
     MP(1,1) GOTO 400
440  REM
450  FOR I = 0 TO J + 1
455  IF X < C(I,0) THEN X1 = I -
     1:I = J + 1
460  NEXT I
462  FOR I = 0 TO K + 1
464  IF Y < D(I,0) THEN Y1 = I -
     1:I = K + 1
466  NEXT I
468  IF X <  > C(X1,0) OR Y <  >
     D(Y1,0) GOTO 477
469  IF S(Y1,X1,0) >  = 0 GOTO 47
     2
```

```
470  IF S(Y1,X1,0) < 0 AND ( SGN
     (S(Y1,X1 - 1,0)) <  > 1) THEN
     X1 = X1 - 1: GOTO 477
471 Y1 = Y1 - 1: GOTO 477
472  IF S(Y1,X1,0) = 0 AND S(Y1,X
     1 - 1,0) = 0 AND X1 < (J / 2
     ) THEN X1 = X1 - 1: GOTO 477

474  IF S(Y1,X1,0) = 0 AND S(Y1 -
     1,X1,0) = 0 AND Y1 < (K / 2)
      THEN Y1 = Y1 - 1
477 CN = 0: IF X = C(X1,0) OR Y =
     D(Y1,0) THEN XA(0) = X:YA(0)
      = Y:CN = 1
478 X2 = X1:Y2 = Y1
480  GOSUB 3000
490  GOSUB 5000
495 CQ = CQ + 1
500 X = XA(1):Y = YA(1):YA(0) = Y
     :XA(0) = X:CN = 1
510  REM  FIND NEW QUADRANT
512 X4 = X1:Y4 = Y1
515  FOR I = 0 TO J + 1
520  IF X <  > C(I,0) GOTO 535
522 F8 = 1
525  IF I = X1 THEN X1 = I - 1:I =
     J + 1: GOTO 535
530 X1 = I:I = J + 1
535  NEXT I
540  FOR I = 0 TO K + 1
545  IF Y <  > D(I,0) GOTO 560
547 F9 = 1
550  IF I = Y1 THEN Y1 = I - 1:I =
     K + 1: GOTO 560
555 Y1 = I:I = K + 1
560  NEXT I
562  IF F8 = 1 AND F9 = 1 THEN  GOSUB
     7000
563 F8 = 0:F9 = 0
565  IF X2 <  > X1 OR Y2 <  > Y1 GOTO
     480
570  GOTO 400
```

```
670  XDRAW CH AT X * XL / XF, (YF -
     Y) * YL / YF
999  STOP
2000  REM  PRINT MIN PT
2010  IF XC > 1 GOTO 2020
2015  PRINT "X OF MIN PT IS ";MP(
     0,0)
2016  GOTO 2025
2020  PRINT "X OF MIN PT IS BETWE
     EN ";MP(0,0);" AND ";MP(1,0)

2025  IF YC > 1 GOTO 2035
2030  PRINT "Y OF MIN PT IS ";MP(
     0,1)
2031  GOTO 2040
2035  PRINT "Y OF MIN PT IS BETWE
     EN ";MP(0,1);" AND ";MP(1,1)

2040  RETURN
3000  REM  FIND ENDPOINTS
3010  IF S(Y1,X1,0) = 0 GOTO 3200

3015  IF D(Y1,0) =  - 1000 OR D(Y
     1,0) = Y GOTO 3035
3020 XT =  - (Y - D(Y1,0) - S(Y1,
     X1,0) * X) / S(Y1,X1,0)
3022  IF  ABS (XT -  INT (XT + .0
     01)) < .002 THEN XT =  INT (
     XT + .001)
3025  IF XT < (C(X1 + 1,0) + .000
     1) AND XT > (C(X1,0) - .0001
     ) THEN XA(CN) = XT:YA(CN) =
     D(Y1,0):CN = CN + 1
3030  IF CN = 2 THEN  RETURN
3035  IF D(Y1 + 1,0) = 1000 OR D(
     Y1 + 1,0) = Y GOTO 3055
3040 XT =  - (Y - D(Y1 + 1,0) - S
     (Y1,X1,0) * X) / S(Y1,X1,0)
3042  IF  ABS (XT -  INT (XT + .0
     01)) < .002 THEN XT =  INT (
     XT + .001)
```

```
3045  IF XT < (C(X1 + 1,0) + .000
     1) AND XT > (C(X1,0) - .0001
     ) THEN XA(CN) = XT:YA(CN) =
     D(Y1 + 1,0):CN = CN + 1
3050  IF CN = 2 THEN  RETURN
3055  IF C(X1,0) =  - 1000 OR C(X
     1,0) = X GOTO 3075
3060 YT = Y - S(Y1,X1,0) * (X - C
     (X1,0))
3062  IF  ABS (YT -  INT (YT + .0
     01)) < .002 THEN YT =  INT (
     YT + .001)
3065  IF YT < (D(Y1 + 1,0) + .000
     1) AND YT > (D(Y1,0) - .0001
     ) THEN XA(CN) = C(X1,0):YA(C
     N) = YT:CN = CN + 1
3070  IF CN = 2 THEN  RETURN
3075 YA(CN) = Y - S(Y1,X1,0) * (X
      - C(X1 + 1,0))
3077  IF  ABS (YA(CN) -  INT (YA(
     CN) + .001)) < .002 THEN YA(
     CN) =  INT (YA(CN) + .001)
3080 XA(CN) = C(X1 + 1,0):CN = CN
      + 1
3082  IF CN < > 2 THEN  PRINT "E
     RROR": STOP
3085  RETURN
3200  REM  COME HERE WITH ZERO SL
     OPE
3210  IF S(Y1,X1,1) < > 0 GOTO 3
     230
3215  IF CN = 0 THEN XA(0) = C(X1
     ,0):XA(1) = C(X1 + 1,0):YA(0
     ) = Y:YA(1) = Y: RETURN
3220 YA(1) = Y: IF X = C(X1,0) THEN
     XA(1) = C(X1 + 1,0): RETURN

3225 XA(1) = C(X1,0): RETURN
3230  IF CN = 0 THEN YA(0) = D(Y1
     ,0):YA(1) = D(Y1 + 1,0):XA(0
     ) = X:XA(1) = X: RETURN
3235 XA(1) = X: IF Y = D(Y1,0) THEN
     YA(1) = D(Y1 + 1,0): RETURN
```

```
3240 YA(1) = D(Y1,0): RETURN
5000  REM  PLOT A LINE
5005 X9(CQ) = XA(0):Y9(CQ) = YA(0
     )
5010  IF XA(0) < 0 OR YA(0) < 0 OR
     XA(1) < 0 OR YA(1) < 0 THEN
      RETURN
5015  IF XA(0) > (XF) OR YA(0) >
     (YF) OR XA(1) > (XF) OR YA(1
     ) > YF THEN  RETURN
5030  HPLOT XA(0) * XL / XF,(YF -
     YA(0)) * YL / YF TO XA(1) *
     XL / XF,(YF - YA(1)) * YL /
     YF
5035  RETURN
6000  REM  PLOT GRID
6005  FOR I = 1 TO NP
6010 XA(0) = PO(I,0):XA(1) = PO(I
     ,0):YA(0) = 0:YA(1) = YF: GOSUB
     5000
6015 XA(0) = 0:XA(1) = XF:YA(0) =
     PO(I,1):YA(1) = PO(I,1): GOSUB
     5000
6020  NEXT I
6025  RETURN
6500  REM  INPUT
6505  PRINT  CHR$ (4);"OPEN TEMP"

6510  PRINT  CHR$ (4);"READ TEMP"

6512  INPUT F$
6513  PRINT  CHR$ (4);"CLOSE"
6515  PRINT  CHR$ (4);"OPEN ";F$
6516  PRINT  CHR$ (4);"READ ";F$
6520  INPUT NP,XF,YF,U$,XG(0),XG(
     1),YG(0),YG(1),XL,YL
6525  FOR I = 1 TO NP
6530  INPUT X,Y,PO(I,2)
6535  GOSUB 6600
6537 CH = I + 32:X = PO(I,0):Y =
     PO(I,1): GOSUB 6700
```

```
6540  NEXT I
6545  PRINT  CHR$ (4);"CLOSE": RETURN

6600  REM  CONVERT TO SCREEN COOR

6605 PO(I,Ø) =  INT (((X - XG(Ø))
      / (XG(1) - XG(Ø)) * XF) + .
     Ø5)
6610 PO(I,1) =  INT (((Y - YG(Ø))
      / (YG(1) - YG(Ø)) * YF) + .
     Ø5)
6612 PO(I,1) = YF - PO(I,1)
6615  RETURN
6700  REM  DRAW CH ON SCREEN
6705  XDRAW CH AT X * XL / XF,(YF
      - Y) * YL / YF
6710  RETURN
7000  REM  FIND NEW QUADRANT ON V
     ERTEX
7005  IF (X4 + 1) > J GOTO 7015
7010  IF C(X4 + 1,Ø) = X THEN X1 =
     X4 + 1: GOTO 7017
7015  IF C(X4,Ø) = X THEN X1 = X4
      - 1
7017  IF (Y4 + 1) > K GOTO 7020
7018  IF D(Y4 + 1,Ø) = Y THEN Y1 =
     Y4 + 1: GOTO 7030
7020  IF D(Y4,Ø) = Y THEN Y1 = Y4
      - 1
7030  IF S(Y1,X1,Ø) = Ø OR ( SGN
     (S(Y1,X1,Ø)) =  SGN (S(Y4,X4
     ,Ø))) OR S(Y4,X4,Ø) = Ø THEN
      RETURN
7035  IF  SGN (S(Y1,X4,Ø)) < >  SGN
     (S(Y4,X4,Ø)) THEN X1 = X4: RETURN

7040 Y1 = Y4: RETURN
7500  REM  GEN PEN COORDINATES
7502 C = Ø
7505  PRINT  CHR$ (4);"PR#3"
7510  PRINT "M1,P,C"
7515  PRINT  CHR$ (4);"IN#3"
```

```
7520  INPUT XS,YS,Z
7522  VTAB 1
7525 Z =  ABS (Z): IF Z >  = 10 THEN
     Z = Z - 10
7530  IF Z = 0 OR Z = 2 GOTO 7537

7535  IF C = 0 GOTO 7520
7536  GOTO 7540
7537  IF C = 1 GOTO 7520
7538 X = XS:Y = YS:C = 1: GOTO 75
     20
7540  PRINT  CHR$ (4);"PR#0"
7545  PRINT  CHR$ (4);"IN#0"
7547 X = (X - XG(0)) / (XG(1) - X
     G(0)) * XF
7548 Y = YF - (Y - YG(0)) / (YG(1
     ) - YG(0)) * YF
7550  RETURN
8000  REM  RETURN TO FACILITY CON
     STRUCT
8010  TEXT : HOME : VTAB 8: INPUT
     "ARE YOU SURE YOU WANT TO EX
     IT (Y OR N)? ";A$
8015  IF A$ <  > "Y" GOTO 400
8017  PRINT  CHR$ (4);
8020  PRINT "RUN BUILD FAC"
8030  END
```

APPENDIX D

USE OF SHAPE TABLES TO DISPLAY ALPHANUMERICS

A shape table is used to store images that can be drawn using the BASIC command

DRAW I AT X,Y

where I is the Ith image in the table and X and Y are the coordinates of the point at which to draw the image. Any image that can be displayed on the screen can be saved as a shape in a shape table. To draw the images the table must be resident in memory, and the program must know the memory location of the shape table. Assuming that the table has been created and saved to a binary file on the disk, it can be loaded using the BLOAD command. The addresses containing the address of the start of the shape table are 46364 for the low address byte and 46365 for the high byte (Apple Computer 1981). After these are loaded into addresses 232 and 233 respectively, the DRAW command will access this shape table.

The program used to generate the shape table used in this study is from Myers(1982). Each alphanumeric character is identified by setting I to the ASCII code of the character minus 32.

APPENDIX E

HARDCOPY GRAPHICS OUTPUT

To create a graphics printout such as Figure 9, it is necessary to have either a graphics printer or a plotter. The software needed to drive the hardware can vary in each instance. This program does not support a hardcopy graphics device. A short program was written to display the contents of the graphics screen after it was saved to a binary file.

Figure 9 was created by running the following program using an Okidata microline 92 printer.

```
10  PRINT  CHR$ (4);"BLOAD CLINE"

20  PRINT  CHR$ (4);"PR#1"
30  POKE  - 16304,0
40  PRINT  CHR$ (9);"G"
50  PRINT  CHR$ (24)
60  POKE  - 16303.0
70  PRINT  CHR$ (9);"8S"
80  PRINT  CHR$ (9):"e"
```

BIBLIOGRAPHY

Anderson, J. D. "Microsoftware for Facility Layout." M.S.Thesis, University of Central Florida, 1982.

Apple, James M. Plant Layout and Material Handling. 3d ed. New York: John Wiley & Sons, 1977.

[Espinosa, Christopher] Apple II Reference Manual. Cupertino, CA: Apple Computer, 1979.

Fitts, P. M., ed. Human Engineering for an Effective Air Transportation and Traffic Control System. Washington,D. C.: National Research Council, 1951.

Foley, J. D., and Van Dam, A. Fundamentals of Interactive Computer Graphics. Reading, MA: Addison-Wesley Publishing Co., 1982.

Francis, Richard L., and White, John A. Facility Layout and Location. Englewood Cliffs, NJ: Prentice-Hall, 1974.

Moore, J. M. Plant Layout and Design. New York: Macmillan Publishing Co., 1962.

Myers, Roy E. Microcomputer Graphics. Reading, MA: Addison-Wesley Publishing Co., 1982.

Shneiderman, Ben. Software Psychology: Human Factors in Computers and Information Systems. Cambridge, MA: Winthrop Publishers, 198Ø.

Applesoft II: BASIC Programming Reference Manual. Cupertino, CA: Apple Computer, [1978].

Graphics Tablet: Operation and Reference Manual. Cupertino, CA: Apple Computer, [1979].

Apple II: The DOS Manual: Disk Operating System. Cupertino, CA: Apple Computer, [1981].